Nearly Optimal Univariate Polynomial Root-finding: Old and New Algorithms

Victor Y. Pan

keywords: Polynomial root-finding, Subdivision, Sparse polynomials, Root-counting, Functional iterations, Deflation, Real roots
2000 Math. Subject Classification: 65H05, 26C10, 30C15

Extended Abstract

We first review the State of the Art and then outline our progress and state some major research challenges. Further details can be found in arXiv:1805.12042

1. The problem and three celebrated approaches. Univariate polynomial root-finding has been the central problem of mathematics and computational mathematics for four millennia, since Sumerian times (see [5], [10], [11]). Interest to it has been revived due to the advent of modern computers and applications to signal processing, control, financial mathematics, geometric modeling, and computer algebra. The problem remains the subject of intensive research. Hundreds of efficient polynomial root-finders have been proposed, and new ones keep appearing (see [6], [7]).

Two known root-finders are nearly optimal. The algorithm of [9] and [13], proposed in 1995 and extending the previous progress in [15] and [8], first computes numerical factorization of a polynomial into the product of its linear factors and then readily approximate the roots.[1] In the case of inputs of large size the algorithm solves both problems of numerical factorization and root-finding in record low and nearly optimal Boolean time, that is, it approximates all linear factors of a polynomial as well as all its roots, respectively, almost as fast as one can access the input coefficients with the precision required for these tasks.[2] The algorithm, however, is quite involved and has never been implemented.

[1]Numerical polynomial factorization has various important applications to modern computations, besides root-finding, in particular to time series analysis, Wiener filtering, noise variance estimation, co-variance matrix computation, and the study of multichannel systems.

[2]For an input polynomial of degree d the bounds on the required input precision and Boolean time are greater by a factor of d for root-finding than that for numerical factorization.

Recently Becker et al in [1] proposed the second nearly optimal polynomial root-finder, by extending the previous advances of [14] and [12] for the classical subdivision iterations. The algorithm has been implemented in 2018 and promises to become practical, but so far the root-finder of user's choice is the package MPSolve, devised in 2000 [2] and revised in 2014 [3]. It implements Ehrlich's iterations of 1967, rediscovered by Aberth in 1973. Currently subdivision root-finder performs slightly faster than MPSolve of [3] for root-finding in a disc on the complex plain containing a small number of roots but is noticeably inferior for the approximation of all roots of a polynomial.[3]

2. Representation of an input polynomial. The algorithms of [9], [13], and [1] involve the coefficients of an input polynomial $p = p(x)$, relying on its representation in monomial basis:

$$p(x) = \sum_{i=0}^{d} p_i x^i = p_d \prod_{j=1}^{d} (x - x_j),\ p_d \neq 0, \tag{1}$$

where we may have $x_k = x_l$ for $k \neq l$. In contrast Ehrlich's and various other functional root-finding iterations such as Newton's and Weierstrass's can be applied to a more general class of *black box polynomials* – those represented by a black box subroutine for their evaluation, e.g., those represented in Bernstein's bases and sparse polynomials such as Mandelbrot's (cf. [2, Eqn.16]).

3. Our progress. Having reviewed the State of the Art, we significantly accelerate subdivision and Ehrlich's iterations by means of properly combining them with known and novel root-finding techniques. Moreover we extend subdivision iterations to black box polynomials, enabling their dramatic acceleration in the case of sparse input polynomials. Next we itemize our progress.

- We dramatically accelerate *root-counting* for a polynomial in a disc on the complex plain, which is a basic ingredient of subdivision iterations.[4]
- Even stronger we accelerate *exclusion test*: it verifies that a disc contains no roots and is the other key ingredient of subdivision iterations.
- We extend our fast exclusion test to *proximity estimation*, that is, estimation of the distance from a complex point to a closest root of $p(x)$.[5]
- We accelerate subdivision iterations by means of decreasing the number of required exclusion tests,
- We accelerate subdivision iterations by means of deflation of small degree factors whose root sets are well-isolated from the other roots of p.
- We accelerate real polynomial root-finding by means of nontrivially extending all our progress with subdivision iterations.

---

[3] The computational cost of root-finding in [9], [13], and [1] decreases at least proportionally to the number of roots in a region of interest such as a disc on the complex plain, while MPSolve approximates the roots in such regions almost as slow as all complex roots.

[4] We count m times a root of multiplicity m.

[5] Proximity estimation for $p'(x)$ is critical in path-lifting polynomial root-finders [4].

- Our simple but novel deflation algorithm supports accurate approximation of all roots of a polynomial of extremely high degree.
- We accelerate Ehrlich's iterations by means of incorporation of the Fast Multipole Method (*FMM*).

4. Further details of root-counting and exclusion test. The previous acceleration of the known root-counting in [1] was justly claimed to be their major algorithmic novelty versus their immediate predecessors of [14] and [12], but we enhance that progress: our root-counting is performed at a smaller computational cost under milder assumptions about the isolation of the boundary circle of a disc from the roots of $p(x)$. We can counter the decrease of the root isolation by a factor of f by means of increasing the number of evaluation points just by a factor of $\log(f)$. Compared to the common recipe of root-squaring this has similar arithmetic cost but avoids coefficient growth. Even if we do not know how well the boundary circle is isolated from the roots we just recursively double the number of evaluation points until correctness of the root count is confirmed. For heuristic confirmation we can stop where the computed root count approximates an integer, and we propose additional verification recipes. The same algorithm enables fast exclusion test for a fixed disc, but by perfroming some simple low cost computations we decrease the need for exclusion tests.

5. Three major research challenges. We hope that our work will motivate further research effort towards synergistic combination of some efficient techniques, both well- and less-known for polynomial root-finding.

Devising practical and nearly optimal algorithms for numerical factorization of a polynomial is still a challenge – both Ehrlich's and subdivision iterations are slower for that task by at least a factor of d than the nearly optimal solution in [9] and [13], which is quite involved and not practically competitive.

Our root-finders accelerate the known nearly optimal ones and promise to become user's choice. Their implementation, testing and refinement are major challenges. This work, just initiated, already shows that our improvement of the known algorithms *is for real*.

References

[1] Becker, R., Sagraloff, M., Sharma, V., Yap, C.: A near-optimal subdivision algorithm for complex root isolation based on the Pellet test and Newton iteration. J. Symb. Comput. **86**, 51–96 (2018)

[2] Bini, D.A., Fiorentino, G.: Design, analysis, and implementation of a multiprecision polynomial rootfinder. Numer. Algs. **23**, 127–173 (2000)

[3] Bini, D.A., Robol, L.: Solving secular and polynomial equations: a multiprecision algorithm. J. Comput. Appl. Math. **272**, 276–292 (2014)

[4] Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and Real Computation. Springer Verlag (1998).

[5] Boyer, C.A.: A History of Mathematics. Wiley, New York (1991)

[6] McNamee, J.M.: Numerical Methods for Roots of Polynomials, Part I, XIX+354 pages. Elsevier (2007) ISBN13: 9780444527295

[7] McNamee, J.M., Pan, V.Y.: Numerical Methods for Roots of Polynomials, Part II, XXI+728 pages. Elsevier (2013) ISBN: 9780444527301

[8] Neff, C.A., Reif, J.H.: An $o(n^{1+\epsilon})$ algorithm for the complex root problem. In: Proc. 35th Ann. IEEE Symp. on Foundations of Comput. Sci. (FOCS '94), pp. 540–547. doi: 10.1109/SFCS.1994.365737

[9] Pan, V.Y.: Optimal (up to polylog factors) sequential and parallel algorithms for approximating complex polynomial zeros. In: Proc. 27th Ann. ACM Symp. on Theory of Computing (STOC'95), 741–750. ACM Press, New York (1995) doi: 10.1145/225058.225292

[10] Pan, V.Y.: Solving a polynomial equation: Some history and recent progress. SIAM Review **39**(2), 187–220 (1997)

[11] Pan, V.Y.: Solving polynomials with computers. American Scientist **86**, 62–69 (1998) doi: 10.1511/1998.1.62

[12] Pan, V.Y.: Approximation of complex polynomial zeros: modified quadtree (Weyl's) construction and improved Newton's iteration. J. Complexity **16**(1), 213–264 (2000) doi: 10.1006/jcom.1999.

[13] Pan, V.Y.: Univariate polynomials: nearly optimal algorithms for factorization and rootfinding. J. Symb. Comput. **33**(5), 701–733 (2002) doi: 10.1006/jsco.2002.0531

[14] Renegar, J.: On the worst-case arithmetic complexity of approximating zeros of polynomials. J. Complex. **3**(2), 90–113 (1987)

[15] Schönhage, A.: The fundamental theorem of algebra in terms of computational complexity. University of Tübingen, Germany (1982)

Victor Y. Pan
Department of Computer Science
Lehman College of the City University of New York

e-mail: `victor.pan@lehman.cuny.edu`